Stock Price Prediction System

SWE3028: Capstone Project

2020312141 Donghun Jung
2019313902 Chanyoung Lee
2018314589 Yujin Seo

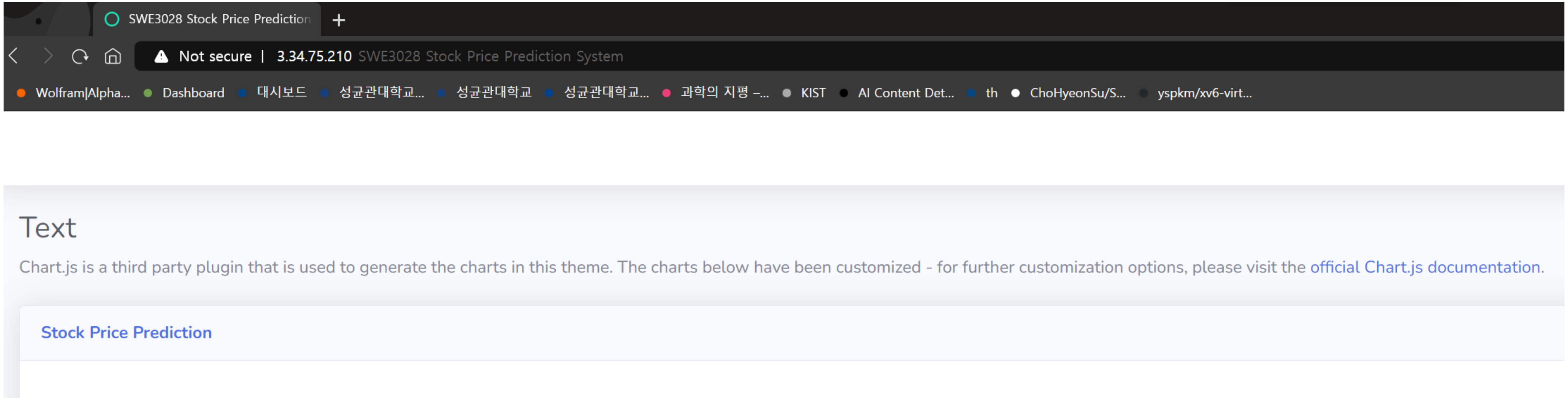
Remind Our Project

Stock Price Prediction System

- Not real time
- Predict the highest and the lowest stock price for each day
- Focus on the short-term price changes

Frontend Implementation

We have deployed our page on AWS server.

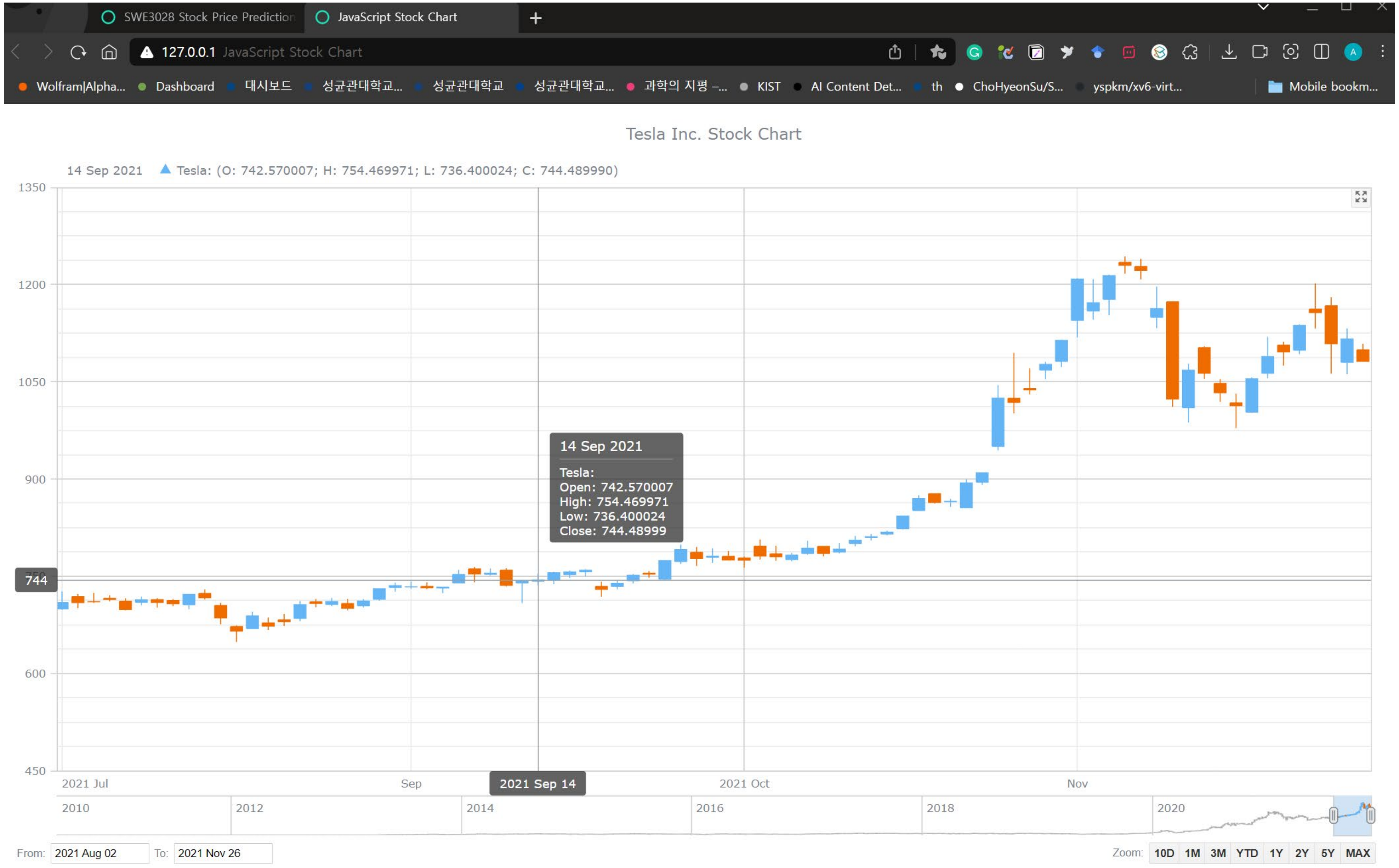


Text

Chart.js is a third party plugin that is used to generate the charts in this theme. The charts below have been customized - for further customization options, please visit the official Chart.js documentation.

Stock Price Prediction

Frontend Implementation



Tesla Inc. Stock Chart

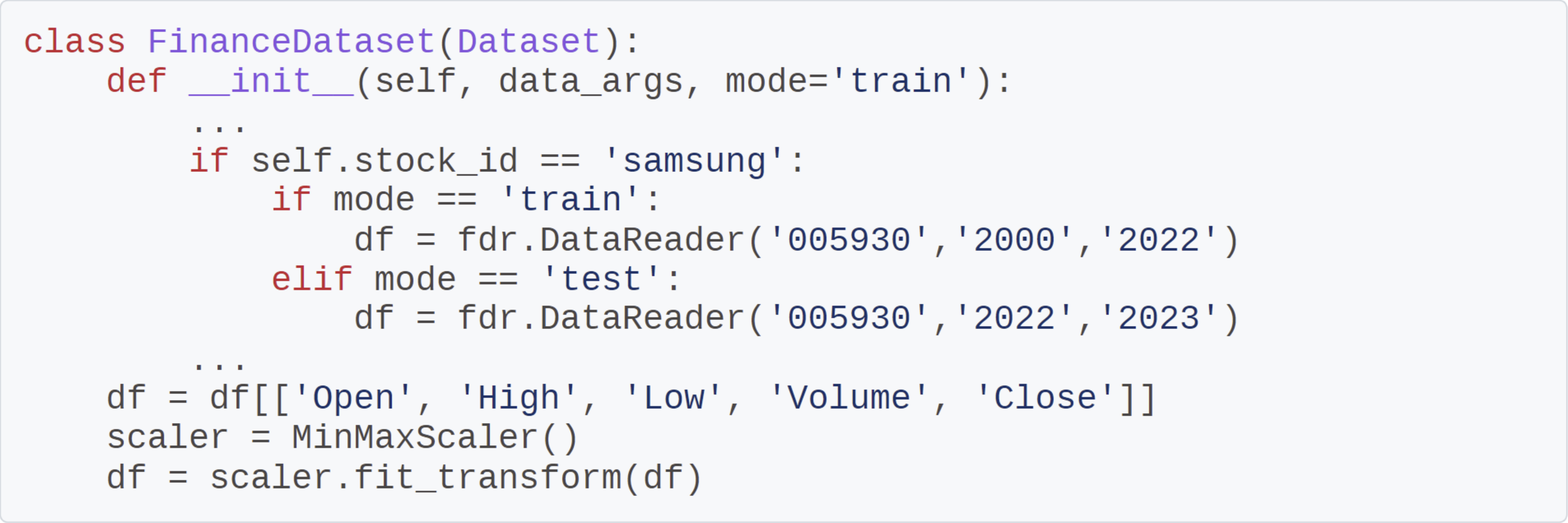
14 Sep 2021 ▲ Tesla: (O: 742.570007; H: 754.469971; L: 736.400024; C: 744.489990)

14 Sep 2021

Tesla:
Open: 742.570007
High: 754.469971
Low: 736.400024
Close: 744.48999

From: 2021 Aug 02 To: 2021 Nov 26

Data

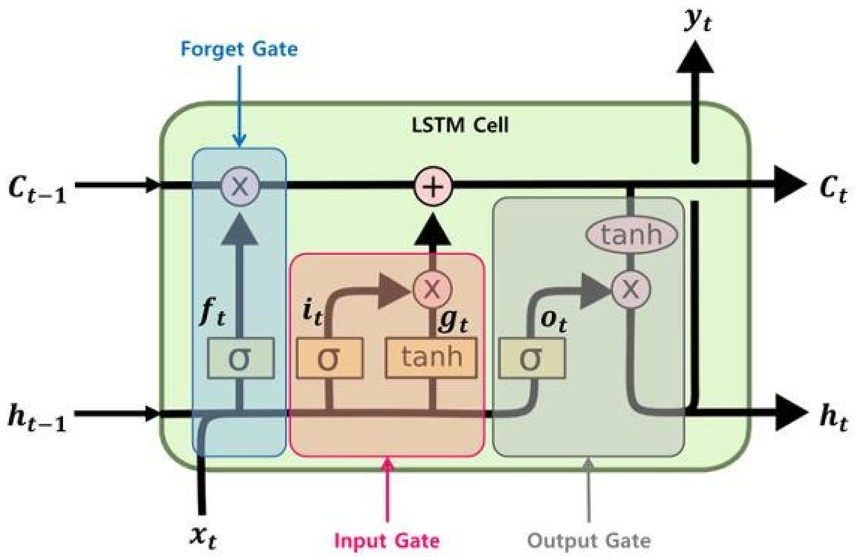


```
class FinanceDataset(Dataset):
    def __init__(self, data_args, mode='train'):
        ...
        if self.stock_id == 'samsung':
            if mode == 'train':
                df = fdr.DataReader('005930','2000','2022')
            elif mode == 'test':
                df = fdr.DataReader('005930','2022','2023')
        ...
    df = df[['Open', 'High', 'Low', 'Volume', 'Close']]
    scaler = MinMaxScaler()
    df = scaler.fit_transform(df)
```

Model Implementation : LSTM

Constructed to treat long and sequential data.

Resolves gradient vanishing problem of RNN when input data is long.

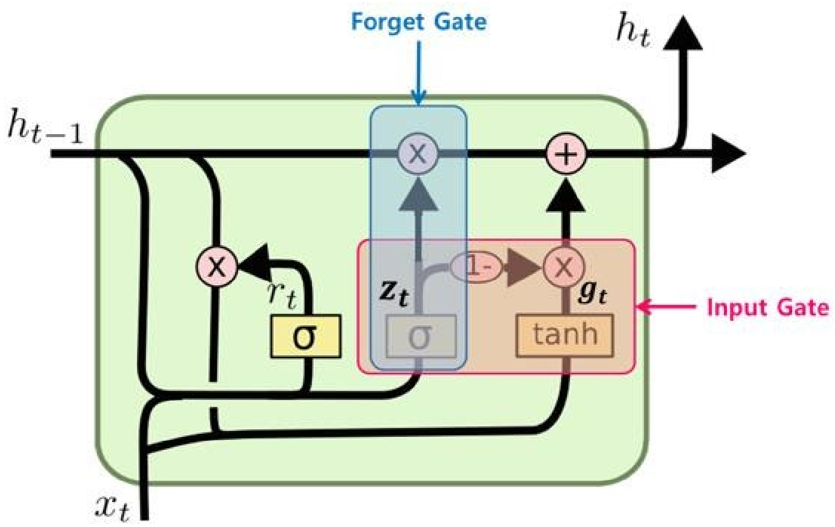


Model Implementation : GRU

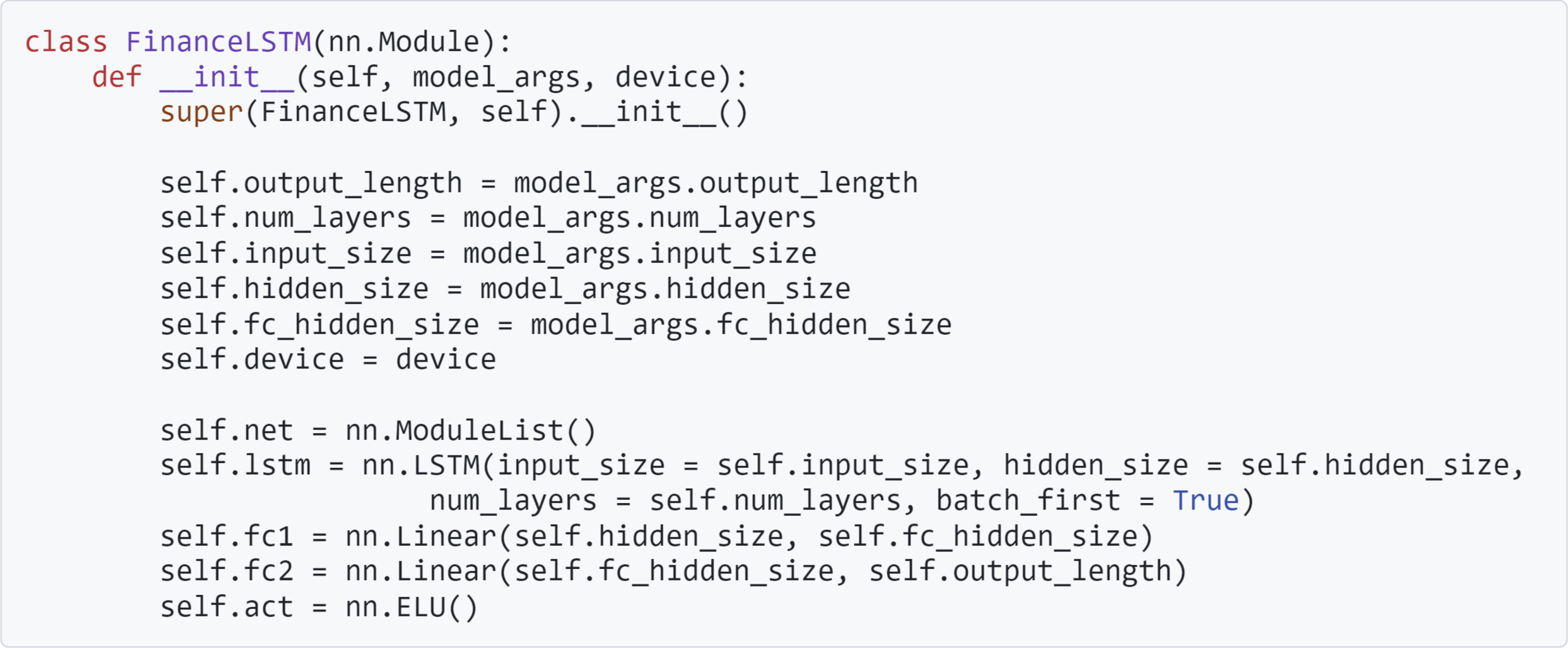
Constructed to improve the performance of LSTM.

Resolves gradient vanishing problem of RNN when input data is long.

GRU has a simpler structure than LSTM.



Model Implementation : LSTM

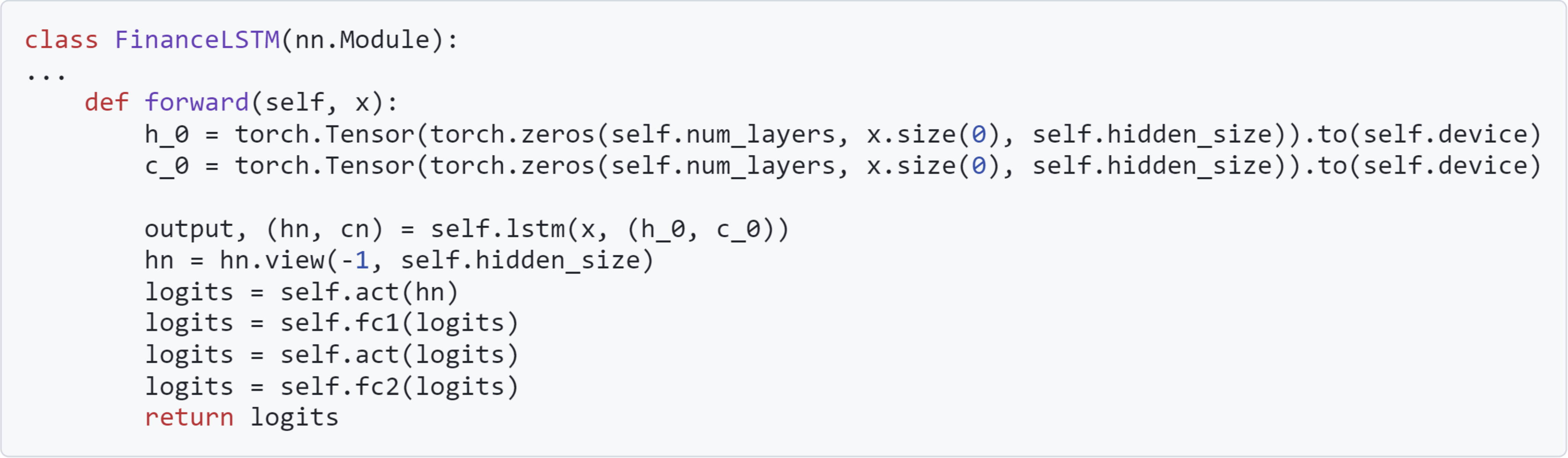


```python
class FinanceLSTM(nn.Module):
    def __init__(self, model_args, device):
        super(FinanceLSTM, self).__init__()

        self.output_length = model_args.output_length
        self.num_layers = model_args.num_layers
        self.input_size = model_args.input_size
        self.hidden_size = model_args.hidden_size
        self.fc_hidden_size = model_args.fc_hidden_size
        self.device = device

        self.net = nn.ModuleList()
        self.lstm = nn.LSTM(input_size = self.input_size, hidden_size = self.hidden_size,
                                num_layers = self.num_layers, batch_first = True)
        self.fc1 = nn.Linear(self.hidden_size, self.fc_hidden_size)
        self.fc2 = nn.Linear(self.fc_hidden_size, self.output_length)
        self.act = nn.ELU()
```

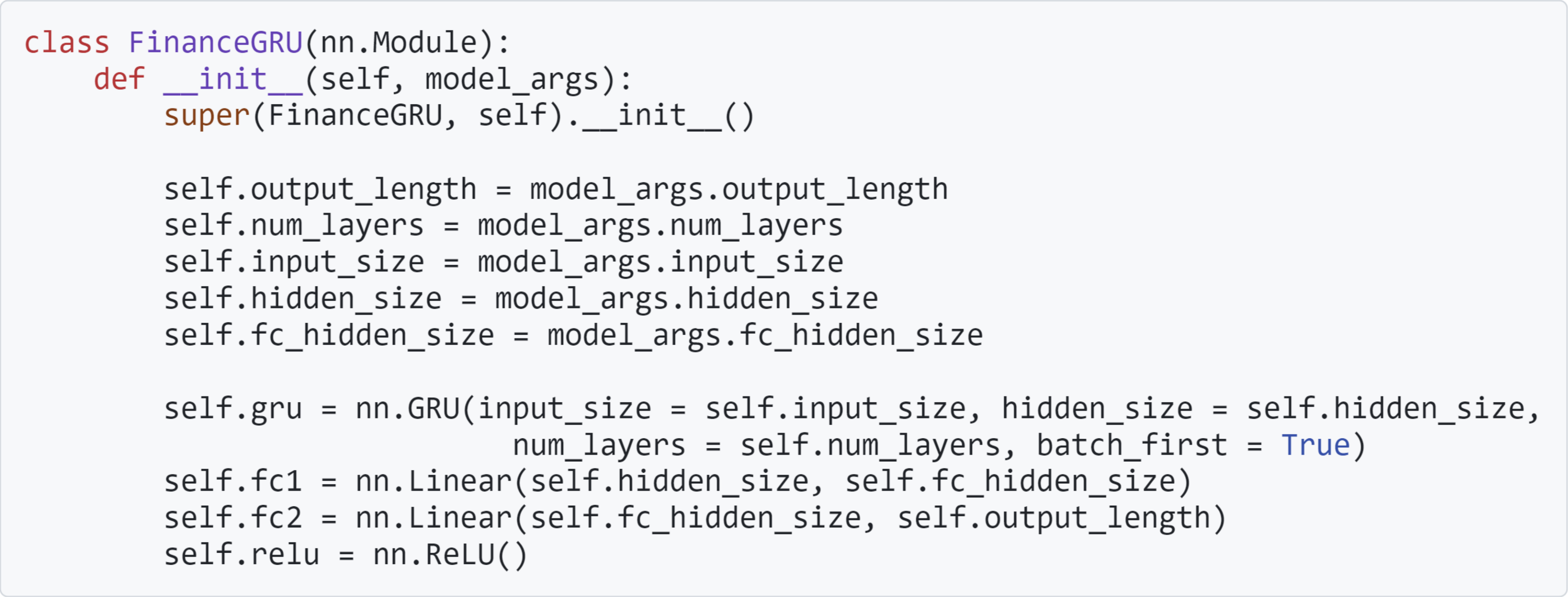
Model Implementation : LSTM



```
class FinanceLSTM(nn.Module):
...
    def forward(self, x):
        h_0 = torch.Tensor(torch.zeros(self.num_layers, x.size(0), self.hidden_size)).to(self.device)
        c_0 = torch.Tensor(torch.zeros(self.num_layers, x.size(0), self.hidden_size)).to(self.device)

        output, (hn, cn) = self.lstm(x, (h_0, c_0))
        hn = hn.view(-1, self.hidden_size)
        logits = self.act(hn)
        logits = self.fc1(logits)
        logits = self.act(logits)
        logits = self.fc2(logits)
        return logits
```

Model Implementation : GRU

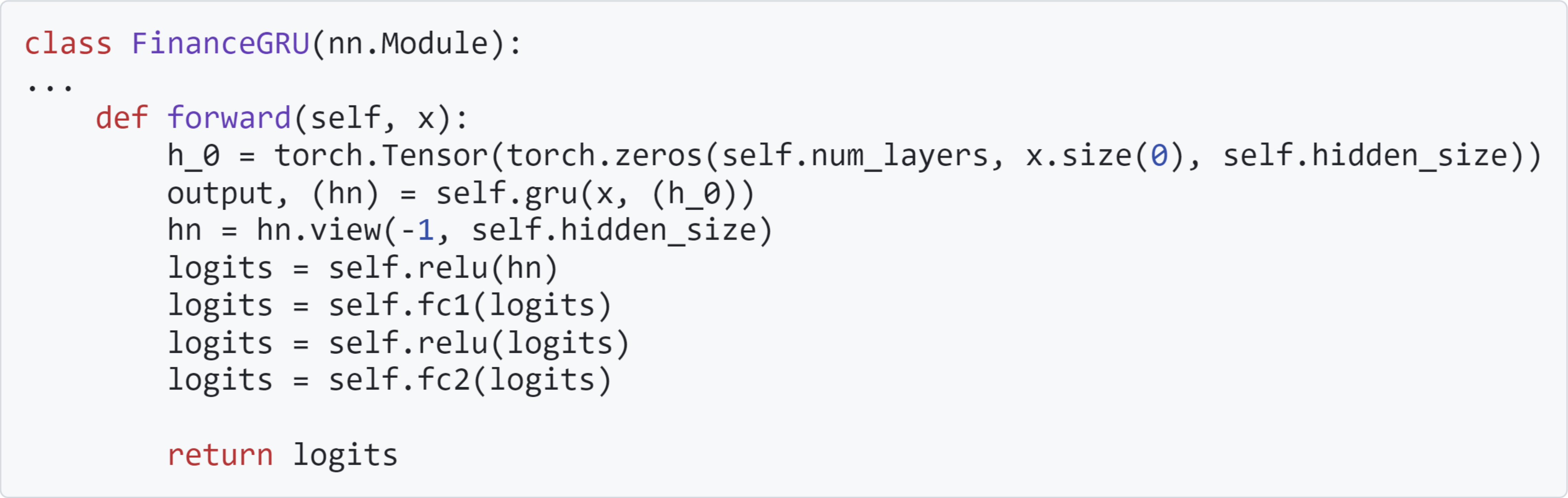


```python
class FinanceGRU(nn.Module):
    def __init__(self, model_args):
        super(FinanceGRU, self).__init__()

        self.output_length = model_args.output_length
        self.num_layers = model_args.num_layers
        self.input_size = model_args.input_size
        self.hidden_size = model_args.hidden_size
        self.fc_hidden_size = model_args.fc_hidden_size

        self.gru = nn.GRU(input_size = self.input_size, hidden_size = self.hidden_size,
                            num_layers = self.num_layers, batch_first = True)
        self.fc1 = nn.Linear(self.hidden_size, self.fc_hidden_size)
        self.fc2 = nn.Linear(self.fc_hidden_size, self.output_length)
        self.relu = nn.ReLU()
```

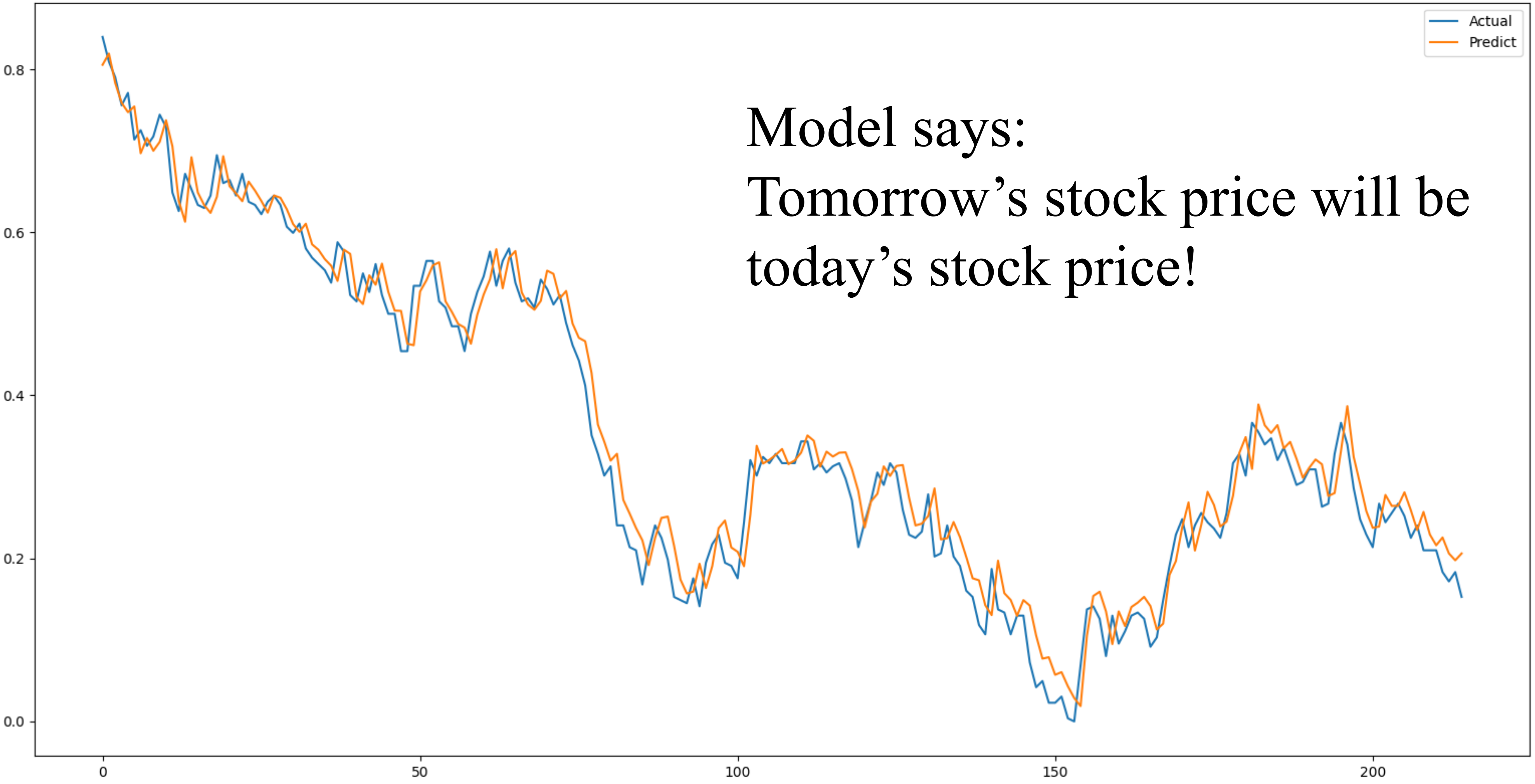
Model Implementation : GRU



```
class FinanceGRU(nn.Module):
...
    def forward(self, x):
        h_0 = torch.Tensor(torch.zeros(self.num_layers, x.size(0), self.hidden_size))
        output, (hn) = self.gru(x, (h_0))
        hn = hn.view(-1, self.hidden_size)
        logits = self.relu(hn)
        logits = self.fc1(logits)
        logits = self.relu(logits)
        logits = self.fc2(logits)

        return logits
```

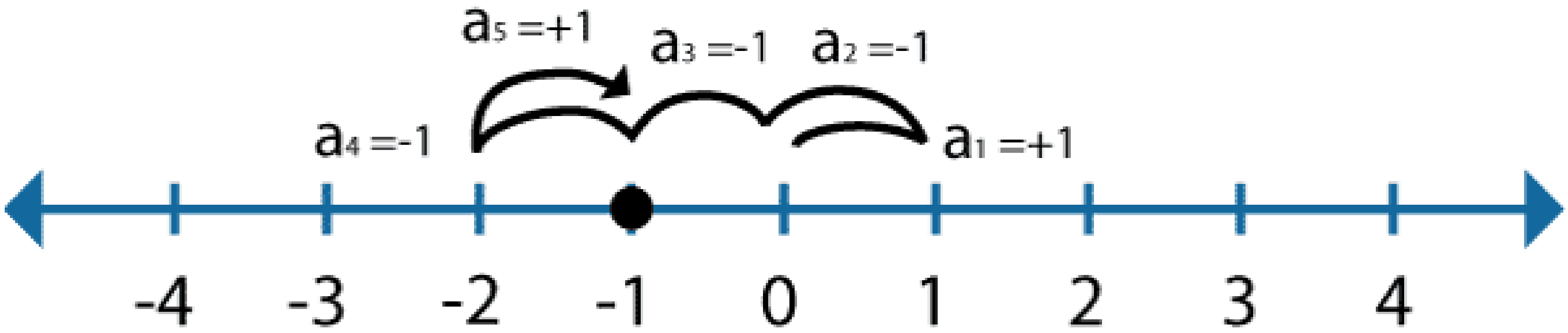
Last Time Result



Model says:
Tomorrow’s stock price will be today’s stock price!

Martingale Theory

The martingale is a sequence of random variables for which, at a particular time, the conditional expectation of the next value in the sequence is equal to the present value, regardless of all prior values.



Model Improvements

1. Data refinement

2. Modify loss function

Model Improvements : Data Refinement

We have redefined the training data.

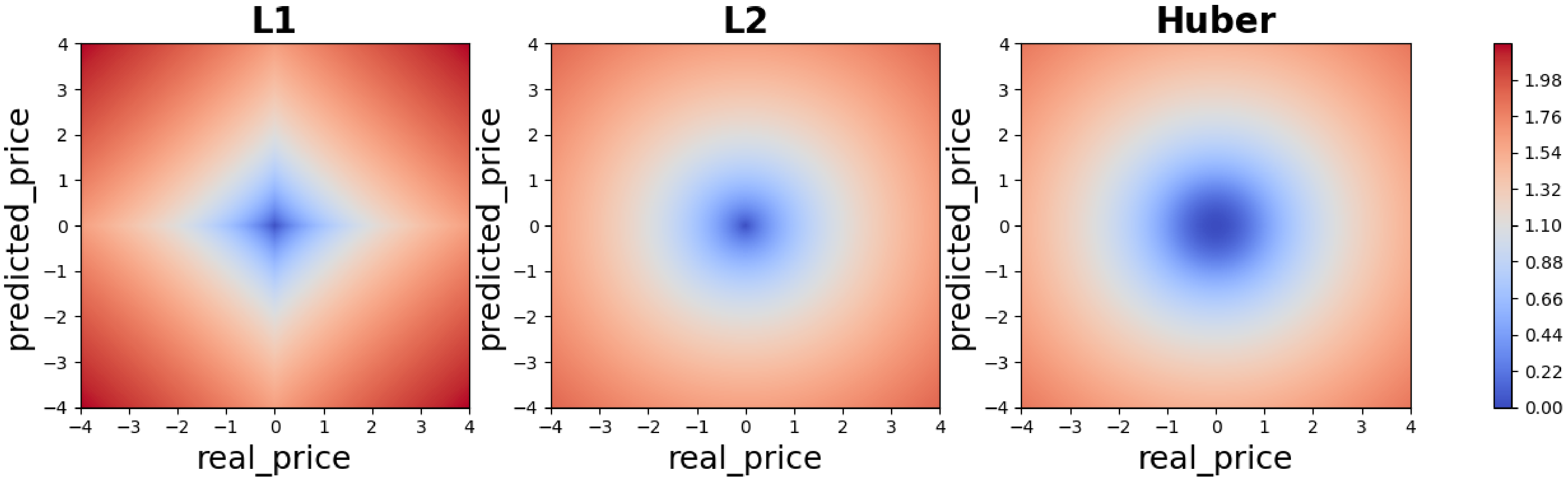
$$\Delta S^{i}(t) = S^{i}(t) - S^{i}(t-1), i \in \{Highest, Lowest\}$$

Model Improvements : Loss Function

$$\mathcal{L} = \mathcal{L}_H + \alpha \sqrt{\sum \left(\frac{input - target}{origin \ - target + \epsilon} \right)^2}$$

$input$: Predicted today price
$target$: Real today price
$origin$: Real yesterday price

$$\mathcal{L}_H = \begin{cases} \frac{1}{2}(input \ - target)^2 & if \ |input \ - target| < \delta \\ \delta \left(|input \ - target| \ - \frac{\delta}{2} \right) & otherwise \end{cases}$$

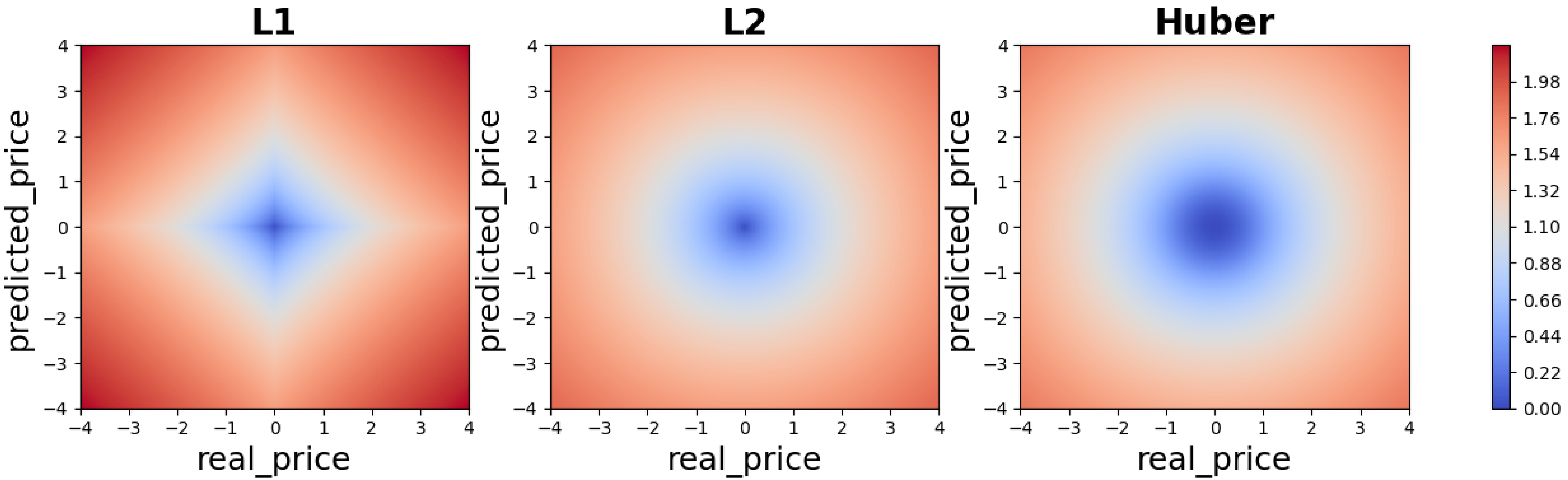


Model Improvements : Loss Function

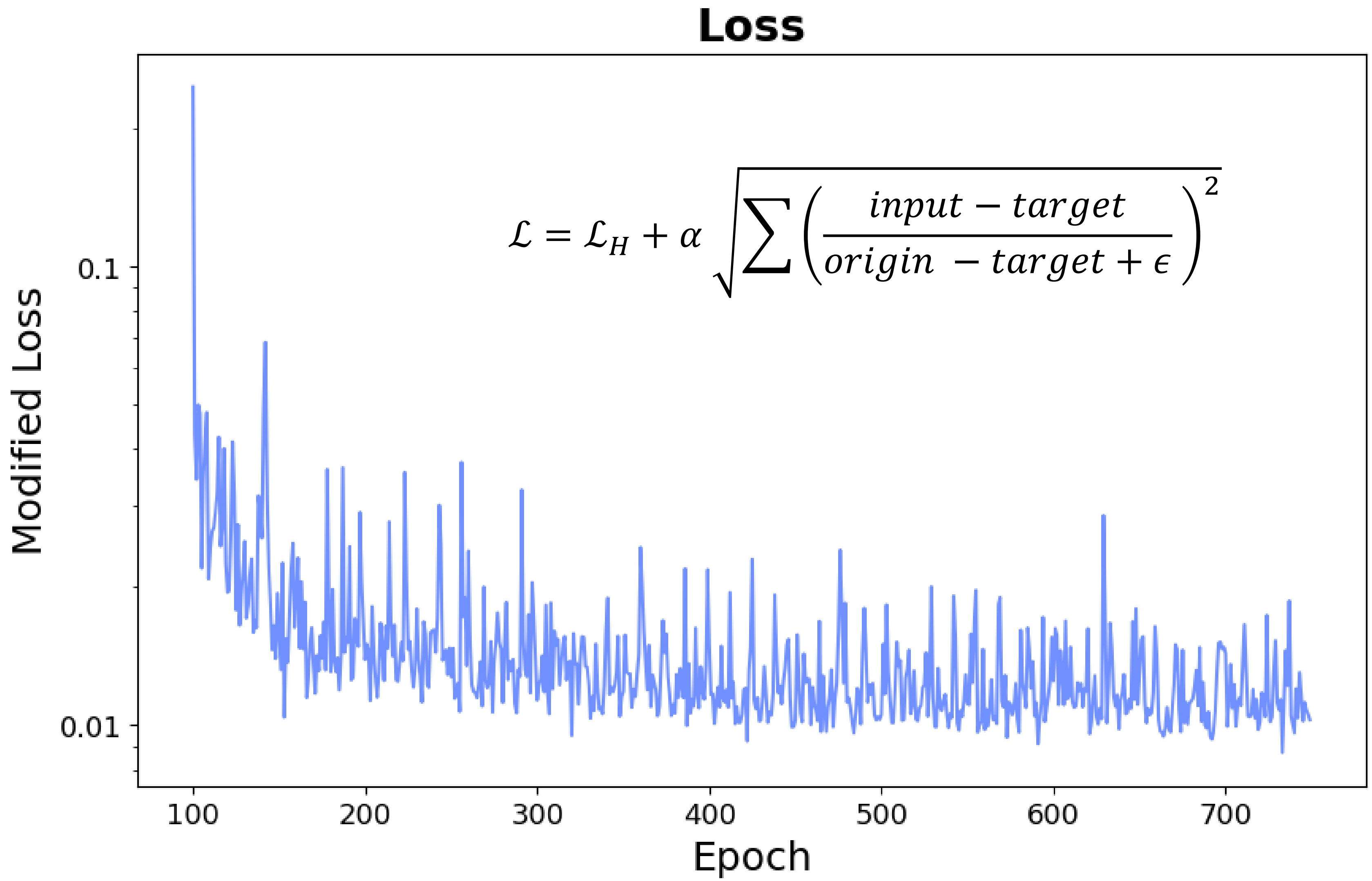
$$\mathcal{L} = \mathcal{L}_H + \alpha \sqrt{\sum \left(\frac{input - target}{origin \ - target + \epsilon} \right)^2}$$

$$\mathcal{L}_H = \begin{cases} \frac{1}{2}(input \ - target)^2 & if \ |input \ - target| < \delta \\ \delta \left(|input \ - target| \ - \frac{\delta}{2} \right) & otherwise \end{cases}$$

$input$: Predicted today price
$target$: Real today price
$origin$: Real yesterday price

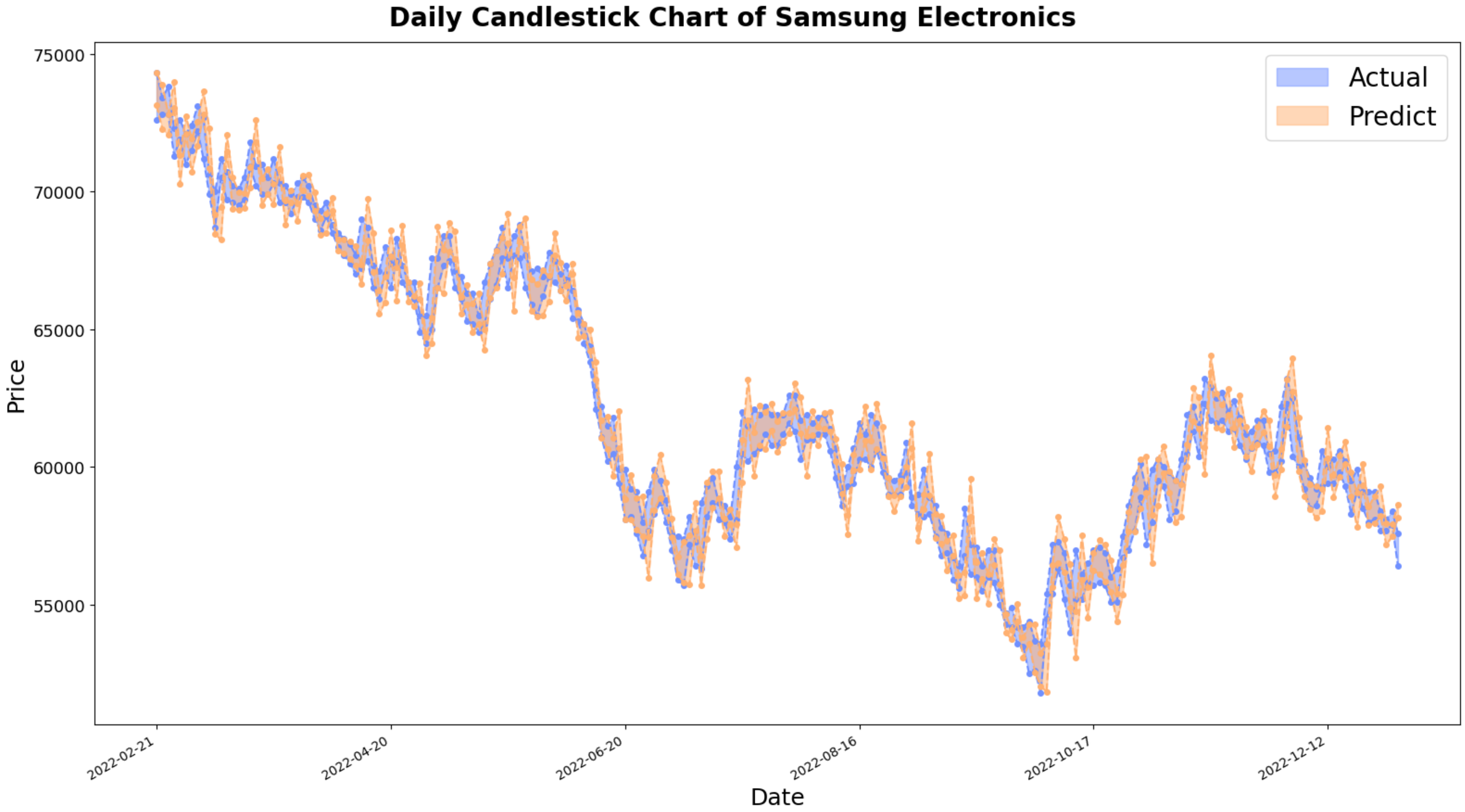


Model Learning curve(LSTM)

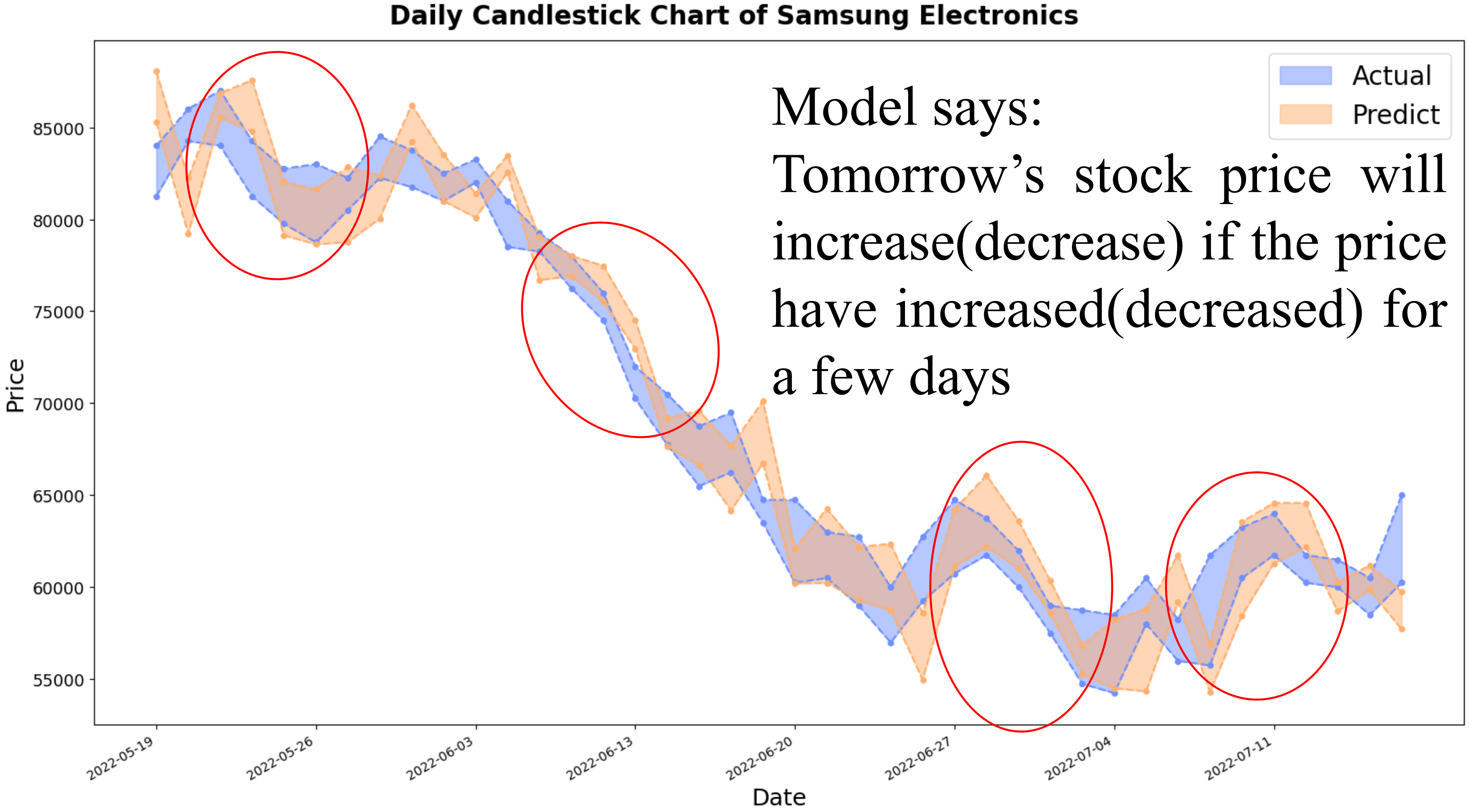


$$\mathcal{L} = \mathcal{L}_H + \alpha \sqrt{\sum \left(\frac{input - target}{origin\ - target + \epsilon}\right)^2}$$

Improvements

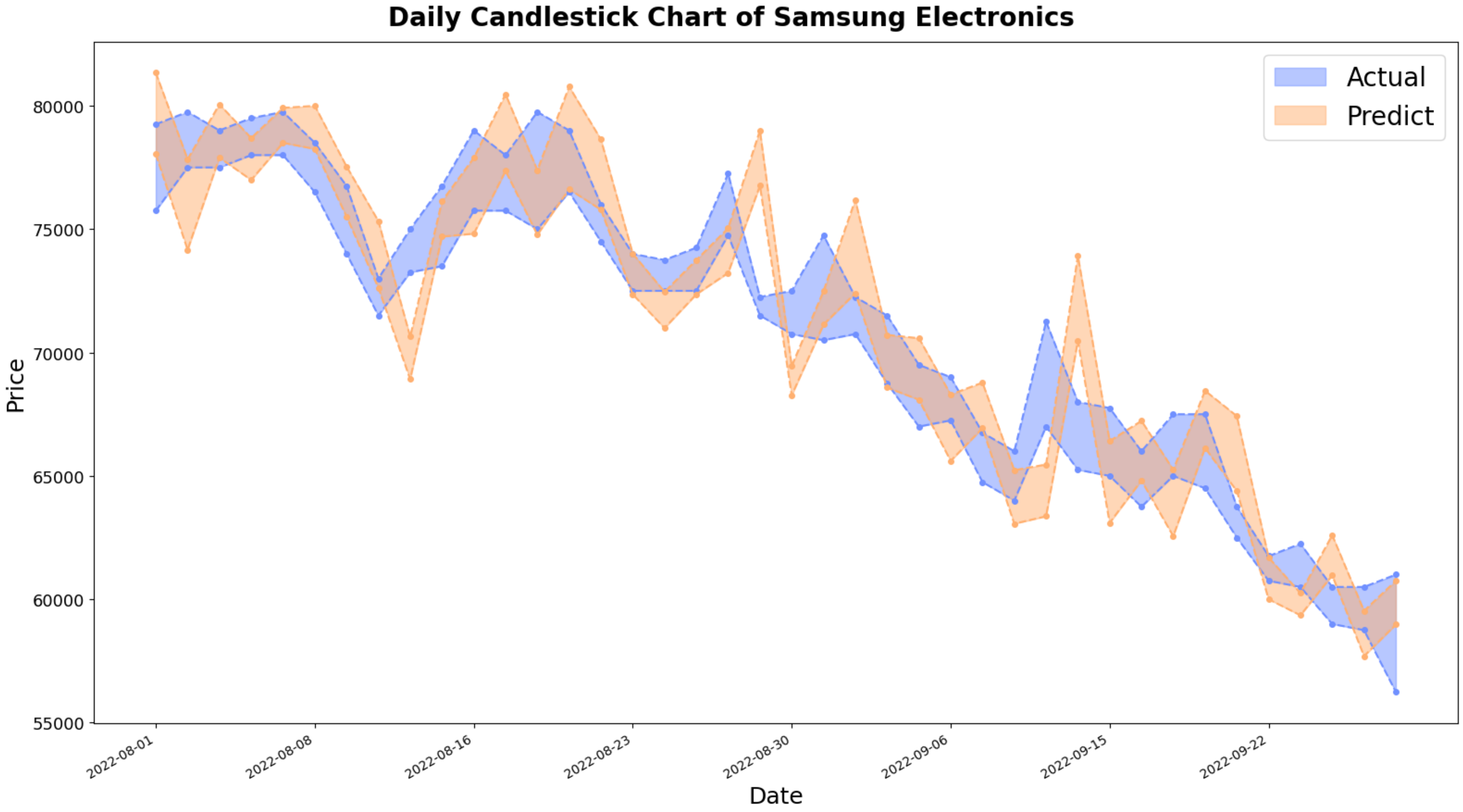


Discussion

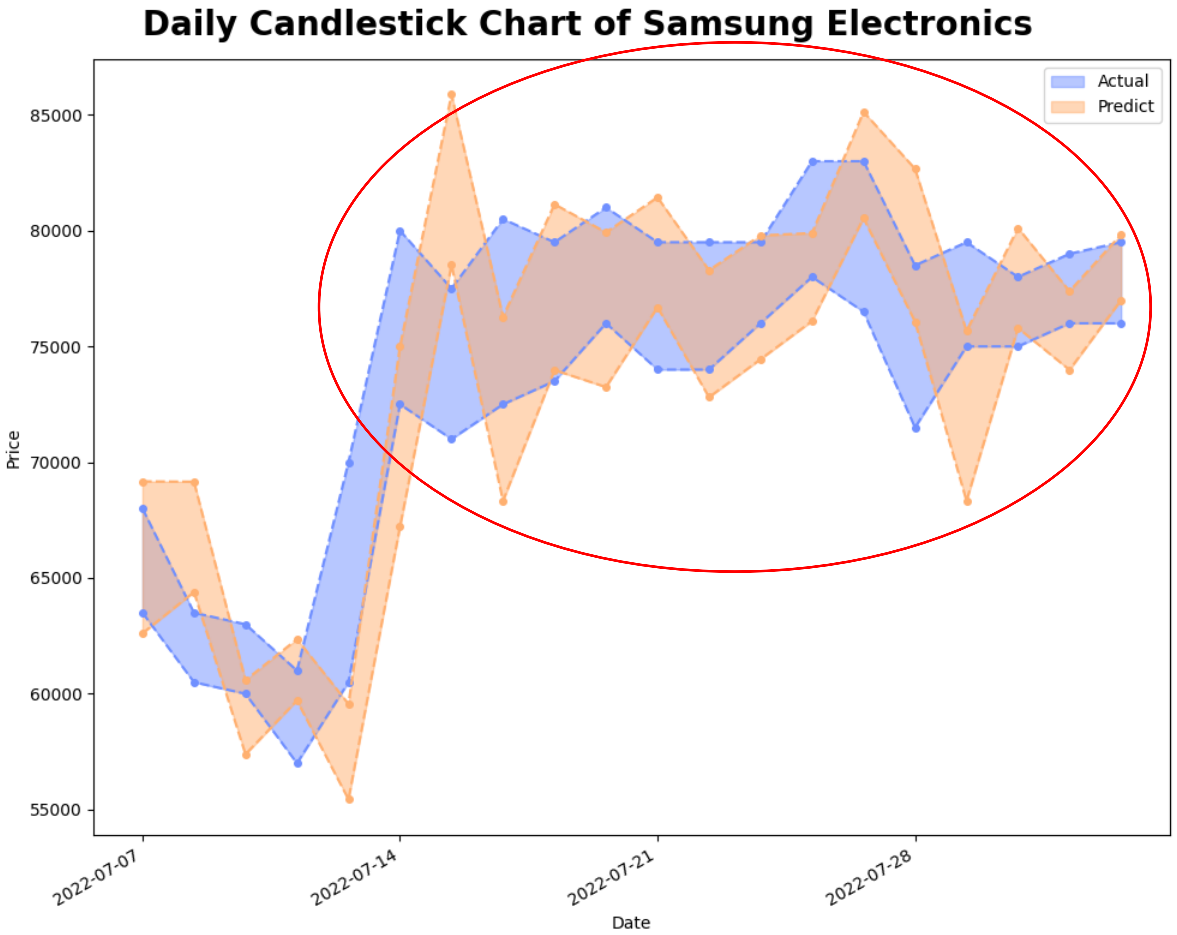


Model says:
Tomorrow's stock price will increase(decrease) if the price have increased(decreased) for a few days

Discussion


Daily Candlestick Chart of Samsung Electronics
Actual
Predict
Price
80000
75000
70000
65000
60000
55000
Date
2022-08-01
2022-08-08
2022-08-16
2022-08-23
2022-08-30
2022-09-06
2022-09-15
2022-09-22

Discussion

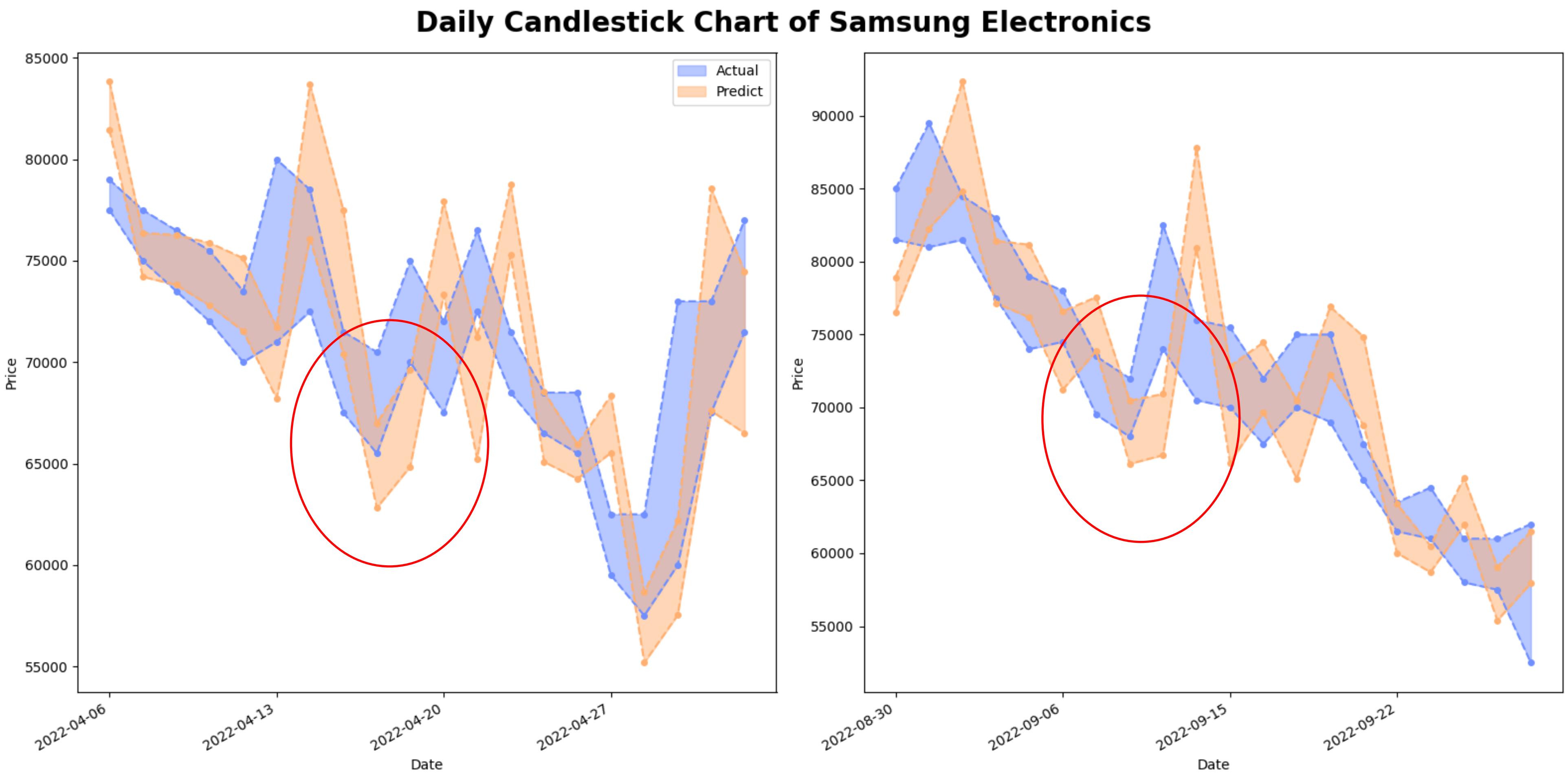


Model says:

Tomorrow's stock price will increase(decrease) if the price have increased(decreased) for a few days

Discussion

But stock price does not keep increasing(or decreasing) and the model knows it!



Summary

1. The model is capable of reading the trend of increasing of decreasing and following it up.

2. The model knows that the stock price occasionally moves in a inverse way.

3. However, it is vulnerable to rapid fluctuating.

Further plan

1. Keep improving models.
   - Hyperparameter tuning
   - Predict more stock prices
2. Evaluation metric : Mean Prediction Accuracy(MPA)

$$MPA = 1 \ - \frac{1}{t}\sum_{t=1}^{T} \frac{|x^t \ - \hat{x}^t \ |}{x^t}$$ (X^t ($\widehat{X^t}$) is a real(prediction) stock price of t-th day)

3. Blended model (LSTM + GRU)
4. Consider some recent issues that may affect on stock price (Vader)

Li, X., Li, Y., Yang, H., Yang, L., & Liu, X. Y. (2019). DP-LSTM: Differential privacy-inspired LSTM for stock prediction using financial news. arXiv preprint arXiv:1912.10806.

Li, Y., & Pan, Y. (2022). A novel ensemble deep learning model for stock prediction based on stock prices and news. International Journal of Data Science and Analytics, 1-11.

Tentative Schedule

	11	12	13	14	15	16
정동훈 Frontend	Build AWS Server UI/UX Design	Update UI/UX Design			Testing	
이찬영 AI	Data Re-define Sentiment analysis	Blended model Vader	Hyperparameter Tuning			
서유진 AI						

Thank you